

APPARATUS AND METHOD FOR TRACE STREAM
IDENTIFICATION OF A PIPELINE FLATTENER
SECONDARY CODE FLUSH FOLLOWING A
RETURN TO PRIMARY CODE EXECUTION

This application claims priority under 35 USC §119(e)(1) of Provisional Application Number 60/434,020 (TI-34666P) filed December 17, 2002.

Related Applications

- 5 U.S. Patent Application (Attorney Docket No. TI-34654),
entitled APPARATUS AND METHOD FOR SYNCHRONIZATION OF TRACE
STREAMS FROM MULTIPLE PROCESSING UNITS, invented by Gary L.
Swoboda, filed on even date herewith, and assigned to the
assignee of the present application; U.S. Patent

5 Application (Attorney Docket No. TI-34655), entitled
APPARATUS AND METHOD FOR SEPARATING DETECTION AND ASSERTION
OF A TRIGGER EVENT, invented by Gary L. Swoboda, filed on
even date herewith, and assigned to the assignee of the
present application; U.S. Patent Application (Attorney
10 Docket No. TI-34656), entitled APPARATUS AND METHOD FOR
STATE SELECTABLE TRACE STREAM GENERATION, invented by Gary
L. Swoboda, filed on even date herewith, and assigned to
the assignee of the present application; U.S. Patent
Application (Attorney Docket No. TI-34657), entitled
15 APPARATUS AND METHOD FOR SELECTING PROGRAM HALTS IN AN
UNPROTECTED PIPELINE AT NON-INTERRUPTIBLE POINTS IN CODE
EXECUTION, invented by Gary L. Swoboda, filed on even date
herewith, and assigned to the assignee of the present
application; U.S. Patent Application (Attorney Docket No.
20 TI-34658), entitled APPARATUS AND METHOD FOR REPORTING
PROGRAM HALTS IN AN UNPROTECTED PIPELINE AT NON-
INTERRUPTIBLE POINTS IN CODE EXECUTION, invented by Gary L.
Swoboda, filed on even date herewith, and assigned to the
assignee of the present application; U.S. Patent
25 Application (Attorney Docket No. TI-34659), entitled
APPARATUS AND METHOD FOR A FLUSH PROCEDURE IN AN
INTERRUPTED TRACE STREAM, invented by Gary L. Swoboda,
filed on even date herewith, and assigned to the assignee
of the present application; U.S. Patent Application
30 (Attorney Docket No. TI-34660), entitled APPARATUS AND
METHOD FOR CAPTURING AN EVENT OR COMBINATION OF EVENTS
RESULTING IN A TRIGGER SIGNAL IN A TARGET PROCESSOR,

5 invented by Gary L. Swoboda, filed on even date herewith,
and assigned to the assignee of the present application;
U.S. Patent Application (Attorney Docket No. TI-34661),
entitled APPARATUS AND METHOD FOR CAPTURING THE PROGRAM
COUNTER ADDRESS ASSOCIATED WITH A TRIGGER SIGNAL IN A
10 TARGET PROCESSOR, invented by Gary L. Swoboda, filed on
even date herewith, and assigned to the assignee of the
present application; U.S. Patent Application (Attorney
Docket No. TI-34662), entitled APPARATUS AND METHOD
DETECTING ADDRESS CHARACTERISTICS FOR USE WITH A TRIGGER
15 GENERATION UNIT IN A TARGET PROCESSOR, invented by Gary L.
Swoboda and Jason L. Peck, filed on even date herewith, and
assigned to the assignee of the present application U.S.
Patent Application (Attorney Docket No. TI-34663), entitled
APPARATUS AND METHOD FOR TRACE STREAM IDENTIFICATION OF A
20 PROCESSOR RESET, invented by Gary L. Swoboda and Bryan
Thome, filed on even date herewith, and assigned to the
assignee of the present application; U.S. Patent (Attorney
Docket No. TI-34664), entitled APPARATUS AND METHOD FOR
TRACE STREAM IDENTIFICATION OF A PROCESSOR DEBUG HALT
25 SIGNAL, invented by Gary L. Swoboda and Bryan Thome, filed
on even date herewith, and assigned to the assignee of the
present application; U.S. Patent Application (Attorney
Docket No. TI-34665), entitled APPARATUS AND METHOD FOR
TRACE STREAM IDENTIFICATION OF A PIPELINE FLATTENER PRIMARY
30 CODE FLUSH FOLLOWING INITIATION OF AN INTERRUPT SERVICE
ROUTINE; invented by Gary L. Swoboda and Bryan Thome, filed
on even date herewith, and assigned to the assignee of the

5 present application; U.S. Patent Application (Docket No. TI-34667), entitled APPARATUS AND METHOD IDENTIFICATION OF A PRIMARY CODE START SYNC POINT FOLLOWING A RETURN TO PRIMARY CODE EXECUTION, invented by Gary L. Swoboda, filed on even date herewith, and assigned to the assignee of the

10 present application; U. S. Patent Application (Attorney Docket No. TI-34668), entitled APPARATUS AND METHOD FOR IDENTIFICATION OF A NEW SECONDARY CODE START POINT FOLLOWING A RETURN FROM A SECONDARY CODE EXECUTION, invented by Gary L. Swoboda, filed on even date herewith,

15 and assigned to the assignee of the present application; U.S. Patent Application (Attorney Docket No. TI-34669), entitled APPARATUS AND METHOD FOR TRACE STREAM IDENTIFICATION OF A PAUSE POINT IN A CODE EXECUTION SEQUENCE, invented by Gary L. Swoboda, filed on even date

20 herewith, and assigned to the assignee of the present application; U.S. Patent Application (Attorney Docket No. TI-34670), entitled APPARATUS AND METHOD FOR COMPRESSION OF A TIMING TRACE STREAM, invented by Gary L. Swoboda and Bryan Thome, filed on even date herewith, and assigned to

25 the assignee of the present application; U.S. Patent Application (Attorney Docket No. TI-34671), entitled APPARATUS AND METHOD FOR TRACE STREAM IDENTIFICATION OF MULTIPLE TARGET PROCESSOR EVENTS, invented by Gary L. Swoboda and Bryan Thome, filed on even date herewith, and

30 assigned to the assignee of the present application; and U.S. Patent Application (Attorney Docket No. TI-34672), entitled APPARATUS AND METHOD FOR OP CODE EXTENSION IN

5 PACKET GROUPS TRANSMITTED IN TRACE STREAMS, invented by Gary L. Swoboda and Bryan Thome, filed on even date herewith, and assigned to the assignee of the present application are related applications.

10 **Background of the Invention**

1. Field of the Invention

15 This invention relates generally to the testing of digital signal processing units and, more particularly, to the signals that are transmitted from a target processor to a host processing to permit analysis of the target processor operation. Certain events in the target processor must be communicated to the host processing unit along with
20 contextual information. In this manner, the test and debug data can be analyzed and problems in the operation of the target processor identified.

2. Description of the Related Art

25 As microprocessors and digital signal processors have become increasingly complex, advanced techniques have been developed to test these devices. Dedicated apparatus is available to implement the advanced techniques. Referring
30 to Fig. 1A, a general configuration for the test and debug of a target processor **12** is shown. The test and debug procedures operate under control of a host processing unit

5 **10.** The host processing unit **10** applies control signals to
the emulation unit **11** and receives (test) data signals from
the emulation unit **11** by cable connector **14**. The emulation
unit **11** applies control signals to and receives (test)
signals from the target processing unit **12** by connector
10 cable **15**. The emulation unit **11** can be thought of as an
interface unit between the host processing unit **10** and the
target processor **12**. The emulation unit **11** processes the
control signals from the host processor unit **10** and applies
these signals to the target processor **12** in such a manner
15 that the target processor will respond with the appropriate
test signals. The test signals from the target processor
12 can be a variety types. Two of the most popular test
signal types are the JTAG (Joint Test Action Group) signals
and trace signals. The JTAG protocol provides a
20 standardized test procedure in wide use in which the status
of selected components is determined in response to control
signals from the host processing unit. Trace signals are
signals from a multiplicity of selected locations in the
target processor **12** during defined period of operation.
25 While the width of the bus **15** interfacing to the host
processing unit **10** generally has a standardized dimension,
the bus between the emulation unit **11** and the target
processor **12** can be increased to accommodate an increasing
amount of data needed to verify the operation of the target
30 processing unit **12**. Part of the interface function between
the host processing unit **10** and the target processor **12** is

5 to store the test signals until the signals can be transmitted to the host processing unit 10.

In testing the target processors, certain events must be identified by the host processing unit. To understand the origin of the program flush sync point, portions of the target processor must be considered in more detail. Referring to Fig. 1B, the target processor pipeline 127 executes program instructions. After the instruction has been processed by the processor pipeline 127, an access of the memory unit 128 results in a delay. To accommodate this delay, the instruction, is placed in a pipeline flattener 129. The pipeline flattener 129 is similar to a first in-first out storage unit. However, the instruction remains in the pipeline flattener 129 until the results of the memory unit access are stored in the location along with the instruction. When the pipeline flattener 129 becomes full, a new instruction results in the transfer from the pipeline flattener 129 to the appropriate location in the target processor.

25 Referring to Fig. 1C, the secondary (interrupt) code execution has been halted or completed (upper graph) and another primary code begins execution (middle graph). The lower graph illustrates that the results of the program execution are being withdrawn from the pipeline flattener. At the breakpoint in the secondary code execution, both the unprotected processor pipeline and the pipeline flattener

5 halt operation. Although instruction results are no longer
being transferred to or from the pipeline flattener, the
results of the memory accesses are still being added to the
instruction locations in the pipeline flattener. After
some period of time, the primary (typically program) code
10 execution begins. As a result of the primary code
execution, the pipeline flattener transfers instruction
results from the secondary code execution before
transferring the results of primary code execution. This
first portion of the output of the pipeline flattener
15 following the beginning of the primary code execution is
designated a "flush" in Fig. 1C. It is important to
communicate to the host processing unit where the flush
portion from the pipeline flattener ends because this point
is actual end point of the secondary code execution prior
20 to initiation of the primary code execution.

A need has been felt for apparatus and an associated method
having the feature that a point at the end of the secondary
code execution flush region is identified in a target
25 processor and that the end point of the flush region is
communicated to the testing apparatus. It is another
feature of the apparatus and associated method to transfer
information concerning a flush region to the testing
processing unit using the trace stream. It is a still
30 further feature of the apparatus and associated method to
communicate to the testing apparatus when the flush region
is identified during the program execution.

5

Summary of the Invention

The aforementioned and other features are accomplished, according to the present invention, by providing the target processor with at least two trace streams. One of the trace streams is a timing trace stream. The second trace stream is the program counter trace stream. When the end of a flush region is identified, a secondary code flush sync marker is generated in the program counter trace stream. This sync marker includes a signal group identifying the event as a secondary code execution flush region, a signal group relating the flush region procedure to the timing trace stream, and a signal group identifying the point in the program execution when the flush region is ended. The point in the program execution where the flush region is identified is determined by indicia in each location in the pipeline flattener indicating when the location is the result of secondary code execution or the result of the primary code execution. When these indicia change for the signal groups being removed from the pipeline flattener, a signal is generated indicating the end of the secondary code flush region. The time of the occurrence of the flush region procedure is determined by trace synchronization markers and by a position of a clock cycle in a timing packet. The trace streams of the target processor are synchronized by periodic sync markers included in the trace streams.

5

Other features and advantages of present invention will be more clearly understood upon reading of the following description and the accompanying drawings and the claims.

10 **Brief Description of the Drawings**

Figure 1A is a general block diagram of a system configuration for test and debug of a target processor, while Fig. 1B is a block diagram illustrating the components of the target processor relevant to the present invention, and Fig. 1C illustrates the operation of the components of Fig. 1B.

Figure 2 is a block diagram of selected components in the target processor used the testing of the central processing unit of the target processor according to the present invention.

Figure 3 is a block diagram of selected components of the illustrating the relationship between the components transmitting trace streams in the target processor.

Figure 4A illustrates format by which the timing packets are assembled according to the present invention, while figure 4B illustrates the inclusion of a periodic sync marker in the timing trace stream.

5 Figure 5 illustrates the parameters for sync markers in the program counter stream packets according to the present invention.

Figure 6A illustrates the sync markers in the program
10 counter trace stream when a periodic sync point ID is generated, while Figure 6B illustrates the reconstruction of the target processor operation from the trace streams according to the present invention.

15 Figure 7 is a block diagram illustrating the apparatus used in reconstructing the processor operation from the trace streams according to the present invention.

Figure 8A is block diagram of the program counter sync
20 marker generator unit; Figure 8B illustrates the sync markers generated in the presence of a secondary code flush condition; Figure 8C illustrates the sync markers in the program counter trace stream in the presence of a secondary code flush condition, and Figure 8D illustrates the
25 reconstruction of the processor operation from the trace stream according to the present invention.

5 Description of the Preferred Embodiment

1. Detailed Description of the Figures

Fig. 1A, Fig. 1B, AND Fig. 1C have been described with
10 respect to the related art.

Referring to Fig. 2, a block diagram of selected components of a target processor **20**, according to the present invention, is shown. The target processor includes at
15 least one central processing unit **200** and a memory unit **208**. The central processing unit **200** and the memory unit **208** are the components being tested. The trace system for testing the central processing unit **200** and the memory unit **202** includes three packet generating units, a data packet
20 generation unit **201**, a program counter packet generation unit **202** and a timing packet generation unit **203**. The data packet generation unit **201** receives VALID signals, READ/WRITE signals and DATA signals from the central processing unit **200**. After placing the signals in packets,
25 the packets are applied to the scheduler/multiplexer unit **204** and forwarded to the test and debug port **205** for transfer to the emulation unit **11**. The program counter packet generation unit **202** receives PROGRAM COUNTER signals, VALID signals, BRANCH signals, and BRANCH TYPE
30 signals from the central processing unit **200** and, after forming these signal into packets, applies the resulting program counter packets to the scheduler/multiplexer **204**

5 for transfer to the test and debug port **205**. The timing packet generation unit **203** receives ADVANCE signals, VALID signals and CLOCK signals from the central processing unit **200** and, after forming these signal into packets, applies the resulting packets to the scheduler/multiplexer unit **204**
10 and the scheduler/multiplexer **204** applies the packets to the test and debug port **205**. Trigger unit **209** receives EVENT signals from the central processing unit **200** and signals that are applied to the data trace generation unit **201**, the program counter trace generation unit **202**, and the
15 timing trace generation unit **203**. The trigger unit **209** applies TRIGGER and CONTROL signals to the central processing unit **200** and applies CONTROL (i.e., STOP and START) signals to the data trace generation unit **201**, the program counter generation unit **202**, and the timing trace
20 generation unit **203**. The sync ID generation unit **207** applies signals to the data trace generation unit **201**, the program counter trace generation unit **202** and the timing trace generation unit **203**. While the test and debug apparatus components are shown as being separate from the
25 central processing unit **201**, it will be clear that an implementation these components can be integrated with the components of the central processing unit **201**.

Referring to Fig. 3, the relationship between selected
30 components in the target processor **20** is illustrated. The data trace generation unit **201** includes a packet assembly unit **2011** and a FIFO (first in/first out) storage unit

5 **2012**, the program counter trace generation unit **202**
includes a packet assembly unit **2021** and a FIFO storage
unit **2022**, and the timing trace generation unit **203**
includes a packet generation unit **2031** and a FIFO storage
unit **2032**. As the signals are applied to the packet
10 generators **201**, **202**, and **203**, the signals are assembled
into packets of information. The packets in the preferred
embodiment are 10 bits in width. Packets are assembled in
the packet assembly units in response to input signals and
transferred to the associated FIFO unit. The
15 scheduler/multiplexer **204** generates a signal to a selected
trace generation unit and the contents of the associated
FIFO storage unit are transferred to the
scheduler/multiplexer **204** for transfer to the emulation
unit. Also illustrated in Fig. 3 is the sync ID generation
20 unit **207**. The sync ID generation unit **207** applies an SYNC
ID signal to the packet assembly unit of each trace
generation unit. The periodic signal, a counter signal in
the preferred embodiment, is included in a current packet
and transferred to the associated FIFO unit. The packet
25 resulting from the SYNC ID signal in each trace is
transferred to the emulation unit and then to the host
processing unit. In the host processing unit, the same
count in each trace stream indicates that the point at
which the trace streams are synchronized. In addition, the
30 packet assembly unit **2031** of the timing trace generation
unit **203** applies an INDEX signal to the packet assembly

5 unit **2021** of the program counter trace generation unit **202**.
The function of the INDEX signal will be described below.

Referring to Fig. 4A, the assembly of timing packets is illustrated. The signals applied to the timing trace
10 generation unit **203** are the CLOCK signals and the ADVANCE signals. The CLOCK signals are system clock signals to which the operation of the central processing unit **200** is synchronized. The ADVANCE signals indicate an activity such as a pipeline advance or program counter advance (())
15 or a pipeline non-advance or program counter non-advance (1). An ADVANCE or NON-ADVANCE signal occurs each clock cycle. The timing packet is assembled so that the logic signal indicating ADVANCE or NON-ADVANCE is transmitted at the position of the concurrent CLOCK signal. These
20 combined CLOCK/ADVANCE signals are divided into groups of 8 signals, assembled with two control bits in the packet assembly unit **2031**, and transferred to the FIFO storage unit **2032**.

25 Referring to Fig. 4B, the trace stream generated by the timing trace generation unit **203** is illustrated. The first (in time) trace packet is generated as before. During the assembly of the second trace packet, a SYYN ID signal is generated during the third clock cycle. In response, the
30 timing packet assembly unit **2031** assembles a packet in response to the SYNC ID signal that includes the sync ID number. The next timing packet is only partially assembled

5 at the time of the SYNC ID signal. In fact, the SYNC ID signal occurs during the third clock cycle of the formation of this timing packet. The timing packet assembly unit **2031** generates a TIMING INDEX 3 signal (for the third packet clock cycle at which the SYNC ID signal occurs) and
10 transmits this TIMING INDEX 3 signal to the program counter packet assembly unit **2031**.

Referring to Fig. 5, the parameters of a sync marker in the program counter trace stream, according to the present
15 invention is shown. The program counter stream sync markers each have a plurality of packets associated therewith. The packets of each sync marker can transmit a plurality of parameters. A SYNC POINT TYPE parameter defines the event described by the contents of the
20 accompanying packets. A program counter TYPE FAMILY parameter provides a context for the SYNC POINT TYPE parameter and is described by the first two most significant bits of a second header packet. A BRANCH INDEX parameter in all but the final SYNC POINT points to a bit
25 within the next relative branch packet following the SYNC POINT. When the program counter trace stream is disabled, this index points a bit in the previous relative branch packet when the BRANCH INDEX parameter is not a logic "0". In this situation, the branch register will not be complete
30 and will be considered as flushed. When the BRANCH INDEX is a logic "0", this value point to the least significant value of branch register and is the oldest branch in the

5 packet. A SYNC ID parameter matches the SYNC POINT with the corresponding TIMING and/or DATA SYNC POINT which are tagged with the same SYNC ID parameter. A TIMING INDEX parameter is applied relative to a corresponding TIMING SYNC POINT. For all but LAST POINT SYNC events, the first
10 timing packet after the TIMING PACKET contains timing bits during which the SYNC POINT occurred. When the timing stream is disabled, the TIMING INDEX points to a bit in the timing packet just previous to the TIMING SYNC POINT packet when the TIMING INDEX value is not zero. In this
15 situation, the timing packet is considered as flushed. A TYPE DATA parameter is defined by each SYNC TYPE. An ABSOLUTE PC VALUE is the program counter address at which the program counter trace stream and the timing information are aligned. An OFFSET COUNT parameter is the program
20 counter offset counter at which the program counter and the timing information are aligned.

Referring to Fig. 6A, a program counter trace stream for a hypothetical program execution is illustrated. In this
25 program example, the execution proceeds without interruption from external events. The program counter trace stream will consist of a first sync point marker 601, a plurality of periodic sync point ID markers 602, and last sync point marker 603 designating the end of the test
30 procedure. The principal parameters of each of the packets are a sync point type, a sync point ID, a timing index, and an absolute PC value. The first and last sync points

5 identify the beginning and the end of the trace stream.
The sync ID parameter is the value from the value from the
most recent sync point ID generator unit. In the preferred
embodiment, this value in a 3-bit logic sequence. The
timing index identifies the status of the clock signals in
10 a packet, i.e., the position in the 8 position timing
packet when the event producing the sync signal occurs.
And the absolute address of the program counter at the time
that the event causing the sync packet is provided. Based
on this information, the events in the target processor can
15 be reconstructed by the host processor.

Referring to Fig. 6B, the reconstruction of the program
execution from the timing and program counter trace streams
is illustrated. The timing trace stream consists of
20 packets of 8 logic "0"s and logic "1"s. The logic "0"s
indicate that either the program counter or the pipeline is
advanced, while the logic "1"s indicate the either the
program counter or the pipeline is stalled during that
clock cycle. Because each program counter trace packet has
25 an absolute address parameter, a sync ID, and the timing
index in addition to the packet identifying parameter, the
program counter addresses can be identified with a
particular clock cycle. Similarly, the periodic sync
points can be specifically identified with a clock cycle in
30 the timing trace stream. In this illustration, the timing
trace stream and the sync ID generating unit are in
operation when the program counter trace stream is

5 initiated. The periodic sync point is illustrative of the plurality of periodic sync points that would typically be available between the first and the last trace point, the periodic sync points permitting the synchronization of the three trace streams for a processing unit.

10

Referring to Fig. 7, the general technique for reconstruction of the trace streams is illustrated. The trace streams originate in the target processor **12** as the target processor **12** is executing a program **1201**. The trace
15 signals are applied to the host processing unit **10**. The host processing unit **10** also includes the same program **1201**. Therefore, in the illustrative example of Fig. 6 wherein the program execution proceeds without interruptions or changes, only the first and the final
20 absolute addresses of the program counter are needed. Using the advance/non-advance signals of the timing trace stream, the host processing unit can reconstruct the program as a function of clock cycle. Therefore, without the sync ID packets, only the first and last sync markers
25 are needed for the trace stream. This technique results in reduced information transfer. Fig. 6 includes the presence of periodic sync ID cycles, of which only one is shown. The periodic sync ID packets are important for synchronizing the plurality of trace streams, for selection
30 of a particular portion of the program to analyze, and for restarting a program execution analysis for a situation wherein at least a portion of the data in the trace data

5 stream is lost. The host processor can discard the (incomplete) trace data information between two sync ID packets and proceed with the analysis of the program outside of the sync timing packets defining the lost data.

10 Referring to Fig. 8A, the major components of the program counter packet generation unit **202** is shown. The program counter packet generation unit **202** includes a decoder unit **2023**, storage unit **2021**, a FIFO unit **2022**, and a gate unit **2024**. PERIODIC SYNC ID signals, TIMING INDEX signals, and

15 ABSOLUTE ADDRESS signals are applied to gate unit **2024**. When the PERIODIC SYNC ID signals are incremented, the decoder unit **2023**, in response to the PERIODIC SYNC ID signal, stores a periodic sync ID header signal group in a predetermined location **2021A** of the header portion of the

20 storage unit **2021**. The PERIODIC SYNC signal causes the gate **2024** to transmit the PERIODIC SYNC ID signals, the TIMING INDEX signals and the ABSOLUTE ADDRESS signals. These transmitted signals are stored in the storage unit **2021** in information packet locations assigned to these

25 parameters. When all of the portions of the periodic sync marker have been assembled in the storage unit **2021**, then the component packets of the periodic sync marker are transferred to the FIFO unit **2022** for eventual transmission to the scheduler/multiplexer unit. Similarly, when a

30 INTERRUPT SERVICE ROUTINE FLUSH signal is generated and applied to the decoder unit **2023**, the interrupt service routine flush header identifying signal group is stored in

5 position **2021A** in the header portion of the storage unit
2021. The INTERRUPT SERVICE ROUTINE FLUSH signal applied
to decoder unit 2023 results in a control signal being
applied to the gate **2024**. As a result of the control
signal, the SYNC ID signals, the TIMING INDEX signals, and
10 the ABSOLUTE ADDRESS signals are stored in the appropriate
locations in storage unit **2021**. When the interrupt service
routine signal sync marker has been assembled, i.e., in
packets, the interrupt service routine flush sync marker is
transferred to the FIFO unit **2022**.

15

Referring to Fig. 8B, the generation of the INTERRUPT
SERVICE ROUTINE signal is illustrated. The pipeline
flattener **129** includes a series of sequential locations
1290. Instruction indicia from the processor pipeline are
20 entered in the pipeline flattener and move in sequence
through the flattener. At some point in the pipeline
flattener **129**, the signals from the related memory unit
access are included with the instruction indicia. Also
included in each instruction indicia is a location subgroup
25 **1291** that stores the origin of the instruction. In Fig.
8B, a P indicates a program code instruction and an I
indicates an interrupt service routine instruction. The
last original subgroup **1291** prior removal of the entire
location signal from the pipeline flattener **129** is applied
30 to decision logic **811**. The decision logic identifies when
the indicia stored in location **1291** changes from the
program (primary) code instruction to an interrupt routine

5 (secondary) code. When the indicia of location changes from an interrupt service routine code to a program code instruction, an event signal is generated and applied to the trigger unit **209**. The trigger unit **209** generates an INTERRUPT SERVICE ROUTINE (or SECONDARY) CODE FLUSH signal
10 and applies this signal to the decoder unit of the **2023** of the program counter trace generation unit **202**.

Referring to Fig. 8C, examples of the sync markers in the program counter trace stream are shown. The start of the
15 test procedure is shown in first point sync marker 801. Thereafter, periodic sync ID markers 805 can be generated. Other event markers can also be generated. The identification of a INTERRUPT SERVICE ROUTINE CODE FLUSH signal results in the generation of the interrupt service
20 routine code flush sync marker 810. Periodic sync ID signals can be generated after the interrupt service routine code flush sync marker and the end of the instruction execution.

25 Referring to Fig. 8D, a reconstruction of the program counter trace stream from the sync markers of Fig. 8B and the timing trace stream is shown. The first sync point marker indicates the beginning of test procedure with a program counter address PC. The program continues to
30 execute unit with the program counter addresses being related to a particular processor clock cycle. When the execution of the pipeline (e.g., an end INTERRUPT SERVICE

5 ROUTINE signal) is generated, the program counter is at
address PC + 9 and is related to a particular clock cycle.
Thereafter, the program counter does not advance as
indicated by the logic "1"s associated with each clock
cycle. Sync ID markers (not shown) can be generated
10 between the first sync point marker and the interrupt
service routine code flush sync marker. At program counter
address PC+10, the primary service routine is initiated.
The instructions with the interrupt service routine code
indicia in the pipeline flattener are removed from the
15 pipeline flattener. This removal identifies the pipeline
flattener latency. When the last interrupt service routine
code instruction is removed from the pipeline flattener,
the INTERRUPT SERVICE ROUTINE CODE FLUSH signal is
generated resulting in a secondary code flush sync marker.

20

2. Operation of the Preferred Embodiment

The present invention relies on the ability of relate the
timing trace stream and the program counter trace stream.
25 This relationship is provided by having periodic sync ID
information transmitted in each trace stream. In addition,
the timing packets are grouped in packets of eight signals
identifying whether the program counter or the pipeline
advanced or didn't advance. The sync markers in the
30 program counter stream include both the periodic sync ID
and the position in the current eight position packet when
the event occurred. Thus, the specific clock cycle of the

5 event can be specified. In addition, the address of the
program counter is provided in the interrupt service
routine flush sync markers so that the interrupt service
routine (secondary) code flush event can be related to the
execution of the target processing unit. Thus, the
10 interrupt service routine code flush sync marker generated
in the program counter trace as a result of the flush of
the pipeline flattener can be related to the target
processor clock, the execution instruction stream of the
target processor, and to the generation of the END
15 INTERRUPT SERVICE ROUTINE signal that resulted in the
change of executing code.

The sync marker trace streams illustrated above relate to an
idealized operation of the target processor in order to
20 emphasize the features of the present invention. Numerous
other sync events (e.g. branch events) will typically be
included in the program counter trace stream.

In the testing of a target processor, large amounts of
25 information need to be transferred from the target
processor to the host processing unit. Because of the
large amount of data to be transferred within a limited
bandwidth, every effort is provided to eliminate necessary
information transfer. For example, the program counter
30 trace stream, when the program is executed in a straight-
forward manner and the sync ID markers are not present,
would consist only of a first and last sync point marker.

5 The execution of the program can be reconstructed as described with respect to Fig. 7. The program counter trace streams includes sync markers only for events that interrupt/alter the normal instruction execution such as branch sync markers and debug halt sync markers.

10

In the foregoing discussion, the sync markers can have additional information embedded therein depending on the implementation of the apparatus generating and interpreting the trace streams. This information will be related to the parameters shown in Fig. 5. It will also be clear that a data trace stream, as shown in Fig. 2 will typically be present. The periodic sync IDs as well as the timing indexes will also be included in the data trace stream. In addition, the program counter absolute address parameter can be replaced by the program counter off-set register in certain situations.

15
20

While the invention has been described with respect to the embodiments set forth above, the invention is not necessarily limited to these embodiments. Accordingly, other embodiments, variations, and improvements not described herein are not necessarily excluded from the scope of the invention, the scope of the invention being defined by the following claims.

25